FILE TRANSFER IN THE DEEP SPACE ENVIRONMENT: ISSUES, DESIRABLE FEATURES, AND PROTOCOL DESIGN

Edward Greenberg

Jet Propulsion Laboratory (JPL)
California Institute of Technology
MS 301-235, 4800 Oak Grove Drive, Pasadena, CA 91109
Fax: (818) 354-9068, E-mail: Edward.Greenberg@jpl.nasa.gov

ABSTRACT

In this paper, the author presents a proposed file transfer protocol tailored to the unique characteristics of deep space missions and their communications networks. The future deep space mission set includes a Mars program that will include multiple orbiters circling Mars acquiring data and providing relay communication support for Mars landed stationary and roving laboratories. The spacelink networks associated with these missions are characterized by (1) zero to infinity (i.e. simplex) response time, (2) non-continuous connections, (3) varying connection path and (4) time disjoint end to end connections. The proposed protocol is designed to operate with connectionless datagram communications and incorporates some unique capabilities required for unmanned mission operations. The provided services include file transfer with selectable levels of quality of service, reliable message delivery and file maintenance and management functionality.

INTRODUCTION

The JPL team supporting the Consultative Committee for Space Data Systems (CCSDS) has been investigating the use of file transfer protocols for near earth and deep space missions. The initial activities focused on methods of extending the Internet protocol suite and lead to the Space Communications Protocol Standards (SCPS) protocol suite. One of these, the SCPS-FP Recommendation, is based on the Internet's File Transfer Protocol (FTP) and effectively provides extensions for Internet operations with near earth spacecraft. The team found that the SCPS-FP lacked many required features and its reliance on a "TCP/IP like connection" is difficult for deep space mission operations.

The present JPL activities are focusing on the issues and features required for cost effective support of future NASA deep space missions including the exploration of Mars missions with their collection of orbiters, landers and surface rovers. The JPL team is directing their file transfer protocol studies toward operations using connectionless datagram communications as provided by the CCSDS Packet Telemetry and Packet Telecommand (in bypass mode) Recommendations. This paper presents the "Reliable Transfer Protocol" (RTP) which is derived from a concept paper presented to the CCSDS.

REQUIREMENTS OVERVIEW

The primary function of the proposed protocol is to provide the following:

- **reliable file transfer** (Example—The mission controllers require reliable file transfer services for the delivery of control instructions and operational data files to the unmanned vehicles as well as the delivery of critical remotely collected and/or highly compressed and processed data that is intolerant of errors or omissions.)
- **incomplete file transfer** (Example—The mission controllers require rapid delivery of selected data much of which provides substantial insight into the performance of the vehicle or the realities of the environment. This insight can often be achieved from incomplete file transfers resulting from simplex operations or when retransmission of missed data is unacceptable due to data latency requirements or over burden the communications links.)
- **reliable application message delivery** (Example—The knowledge of the delivery of selected messages is important to mission controllers. The message could inform the application of the files arrival and instruct the user what to do with it or it could be a message for which the controller want to receive an acknowledgment of its arrival.)
- **file maintenance and file management** (Example—The end to end management of of data trafficking within these missions is simplified by the use of file. It thus becomes important for the system to provide file maintenance and file management services (such as delete and rename a file or get a directory listing) to accomplish these task remotely.

Some of the additional required capabilities are:

- **no handshaking required to initiates a file transfer** (Example—the planetary mission controller has cognizance of the resources available and the authority to send a file. Communication delays caused by the long distances would greatly reduce throughput efficiency if handshaking were required.)
- **efficient use of processor memory and filestore resources** (Examples—the user application creates the file placing it in the filestore, the RTP agent responding to a user's request accesses the file segment by segment and sends it to a remote recipient. Retransmissions are regenerated from the file directly and thus do not require the processor to buffer them while waiting for the acknowledgment.)
- **operate over unreliable communications links** (Example—segments of file data may be missed or delivered out of order.)
- **provide link by link reliability for operations with time disjoint links** (Example—orbiters communicate with earth based stations at times other than when they communicate with landers or rovers)
- **provide end to end accounting where data transfer is shared by multiple intermediaries** (Example—where multiple orbiters or earth stations are needed to transfer a single file of user application data to the remote user)
- **provide for the transfer of custody of a file to allow efficient filestore utilization** (Example—the rover could delete the file from its filestore once acknowledged receipt is obtained from the orbiter as the file traverses the series of links to the end user.)
- **limit latency within an intermediate store and forward node** (Example—when the spacelink data rate is very low and the user needs to review the data as it arrives and before completed)
- **provide persistent sessions** (Example—tracking pass periods may be insufficient for the reliable transfer of an entire file)

- **provide a file capture services for science instruments** (Example—science instruments processors may not be networked to the filestore and can only delivery their data as a stream of packets.)
- **supports multiple concurrent file transfers** (Example—because of the long round trip light time delays a spacecraft could completely transmit a file and start sending another file before the receiving station could acknowledge completeness or the need for the retransmission of missed elements within the first file, thus both transfers need be open simultaneously)
- **provide remote users with same functionality afforded local users** (Example—the mission data management function could request a directory listing from a remote spacecraft and then based on that listing could generate requests for transfer of selected files to designated destinations)

RTP AGENT ARCHITECTURE

The basic architecture of an RTP agent is shown in figure 1. The RTP agent is composed of four internal entities. The **"dispensing entity"** performs the operations associated with sending data to a remote RTP and interfacing with the local filestore for file transfer transactions. The **"acquiring entity"** performs the operations associated with receiving data from a remote RTP agent and assembles the files within the local filestore. The **protocol data unit (PDU) handler** provides the interface with the underlying spacelink and/or ground communication service routing the received RTP PDUs to the appropriate RTP entity. The **RTP manager** entity interfaces with the local user, overviews the transfer functions and interfaces with the filestore performing the required file management tasks.

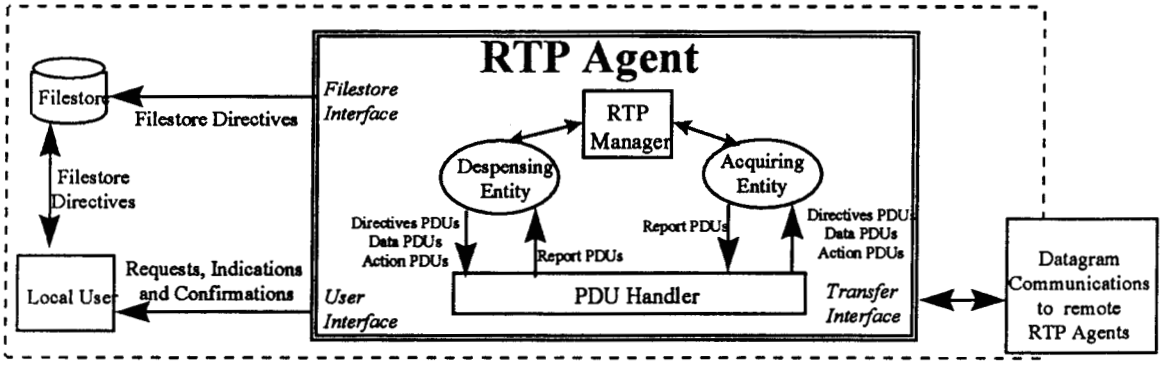


Figure 1 RTP Agent Architecture

THE PROTOCOL

The RTP incorporates session and transport functionality to provide efficient operations using the connectionless datagram communications provided by the CCSDS Packet Telemetry and Packet Telecommand (in bypass mode) Recommendations. A session within the RTP is associated with the transfer of a file or an application message or a file management directive. The session is called a "transaction" and contains a globally unique identifier which enables a receiving RTP agent to merge data independent of its path. The transaction is initiated by the RTP agent at the behest of a local or remote user. The originating RTP agent creates the globally unique transaction identifier by incorporating its own identity with a sequence number it controls. The transaction

identifier will then accompany each protocol data unit (PDU) associated with the transaction within any and all communications links. The accounting for a transaction is performed and maintained within the RTP agents and does not rely on or require the supporting communication's service to provide complete and in order data delivery. The RTP agent uses the accounting information to determine completeness, for requesting data retransmissions, for informing a recipient application of completeness when incomplete file transfer is involved and for closing/terminating transactions.

The basic service interface operates as follows:

- the user sends a request to the RTP agent (e.g. send a new or replacement file, send a request for a directory listing). The request contains the metadata parameters required by the RTP agent for the execution of the transfer. These parameters contain the information to identify the basic quality of service desired, the recipient, and the file attributes. An end to end acknowledgment of completion can be requested from the recipient
- an indication of the receipt of the request is returned to the requester containing the assigned transaction identifier for reference by the requestor.
- when the transaction is initiated an indication is provided to the user if desired.
- the user can cancel the request (even after it is initiated).
- the user can request status reports from the RTP agent.
- a notification of transfer completion to the destination or to a neighboring RTP agent is a provided capability

The following services are provided:

- transfer a file to a designated RTP agent and deliver an associated message to the recipient user application.
- deliver an application message (without a file) to a designated recipient application.
- cause a file management function to be executed by the remote RTP agent. The functions supported are create or delete a directory, delete or rename a file, or create and send a directory list to the designate recipient.

The protocol defines four distinct types of protocol data units (PDU) for the communications between the RTP agents. The content for each PDU type is shown in figure 2. The Metadata, Data and Action PDUs are issued by the dispensing entity. The Report PDU is issued by the acquiring entity to inform the dispensing entity on the status of the transaction. The **Metadata PDU** transports the parameters required to identify the type of function is to be performed (e.g. replace a file or delete a file), to specify the optional features to include (e.g. use simplex delivery and request user to be inform upon arrival of each segment of the file), to identify the required file attributes (e.g. local file name and path, destination file name and path.) The **Data PDU** contains a segment of file data that is being transferred from the dispensing entity's filestore to the acquiring entity's filestore. The **Action PDU** contains directives for control of the transaction along with associated parameters. Examples of actions associated with the Action PDU include the directive to cancel the transaction or to request a transaction status report. **Report PDUs** are issued by the acquiring entity to inform the dispensing entity about the status of the transaction. This PDU may contain the information describing the data that has been received, or a notification of completion or cancellation or rejection of the transaction.

PDU-types	Metadata PDU	Data PDU	Action PDU	Report PDU
Parameters contained within each PDU type	PDU-Identifier	PDU-Identifier	PDU-Identifier	PDU-Identifier
	Report Flag	Report Flag	Report Flag	Relay Report Flag
	Silence Flag	Silence Flag	Silence Flag	Metadata Ack Flag
	Transaction ID	Transaction ID	Transaction ID	Transaction ID
	PDU Size Field	PDU Size Field	PDU Size Field	PDU Size Field
	Subtype Identifier	Offset Pointer	Subtype Identifier	Subtype Identifier
	TLV Field	File Segment Field	TLV Field	TLV Field

Figure 2. Protocol Data Unit types and their basic parameters

PDU Parameter Definitions:

- The **PDU Identifier** identifies the PDU's functional type.
- The **Report Flag** signals the acquiring entity that a status report is desired from the acquiring entity.
- The **Relay Report Flag** informs the dispensing entity to relay this report to the transaction's originating dispensing entity as identified within the TID.
- The **Silence Flag** informs the acquiring entity not to send reports for this transaction unless specifically directed to by an Action PDU.
- The **Metadata Ack Flag** indicates that the acquiring entity has received the metadata associated with this transaction.
- The **Transaction ID** uniquely identifies the transaction to which the PDU pertains. This field has three sub-fields: 1) the originating source RTP agent's ID, 2) the recipient RTP agent's ID and 3) a monotonically ascending sequence number which in combination with the source RTP ID uniquely identifies the transaction for a limited lifetime.
- The value in the **PDU Size Field** contains the number of octets in the PDU.
- The **Sub Type Identifier** identifies the specific function addressed by this PDU. Each of the PDU types have a list of specific function *(Note that it is anticipated that in the future, functionality of the protocol may be extend to include new services)*
- The numeric value contained in the **Offset pointer** identifies where within the file the contained data should be placed.
- The **File Segment field** contains a portion of the transferred file's data.
- The **Type_Length_Value Fields** are composed of self identified and self delimited parameters. The first octect of the field identifies the parameter , the next octect specifies the parameter value's length in octets, and the value itself immediately follows. The TLV field in the Metadata PDU would contain parameters like local file name and path, remote file name and path, file size, file CRC, the recipients agent and application identifiers, when to notify the recipient, and a message to deliver to the recipient. The parameters in the TLV field of an Action PDU could contain cancel or notice that no more data will be sent until report is received. The Report PDU TLV parameters would contain summary information as to the progress of the transaction, what was received or missing, and whether the recipient will participate in the transfer or denies or terminates transaction.

BASIC OPERATIONAL SCENARIO

The user submits a transaction request to the RTP manager. The input is examined by the manager to identify the select optional features and determine its priority for delivery. The manager assigns a transaction identifier and returns an indication. Since network connectivity is discontinuous the manager must rely on supporting capabilities to determine when the correct communications links is or is not to be available. Once the conditions are met the transaction is initiated, the "dispensing entity" then formulates the Metadata PDU containing the metadata required by the acquiring entity. When a file transfer is included the dispensing entity accesses the file from the file store and segments the files as required into Data PDUs for transfer across the communications link to the remote RTP's "acquiring entity." The protocol will retransmit a segment of the file or the Metadata PDU when notified of its loss or optionally when no notice of its acceptance is received within the allowed time window. The acquiring entity will respond as required to the dispensing entity to accomplish and when required to confirm the transfer. The acquiring entity will locally store a transaction's metadata and alert its manager to the incoming data transfer. The RTP manager associated with the acquiring entity will decide whether to accept or reject the transfer based on such things as storage available, use access rights, and file existence conditions. A rejection will initiate a response to the dispensing entity and all following PDUs associated with the transaction will be ignored. If the transfer is accepted, the acquiring entity will reassemble the file in its subdirectory within the local filestore. Upon completion of the transfer the acquiring entity will inform the manager who will then rename the file as require and move it to the appropriate directory. Once accomplished the user will be informed and the associated metadata message will be delivered. If the file is not completed, and the directive includes a request for delivery even when complete transfer could not be accomplished, a the metadata message along with a pointer to the file and the files status will be delivered.

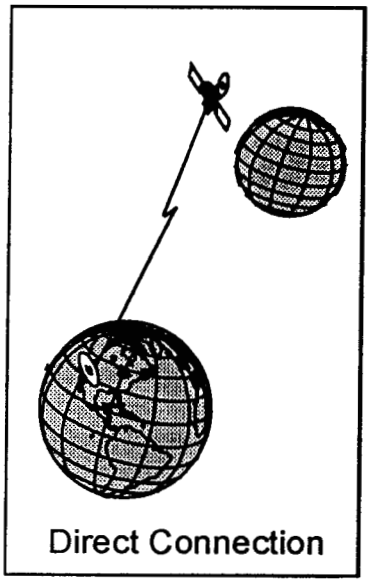
DIRECT CONNECTION SCENARIO
(point-to-point)

When the final destination is connected via a communications link or of a series of links utilizing a network datagram service the operations takes place as described above in the Basic Operational Scenario. The link may be discontinuous as would occur when the contact time associated with a tracking pass is inadequate to complete the transfer of a large file. The RTP agent would keep the session open and reinitiate it when contact is resumed. This is portrayed in scenario picture 1.

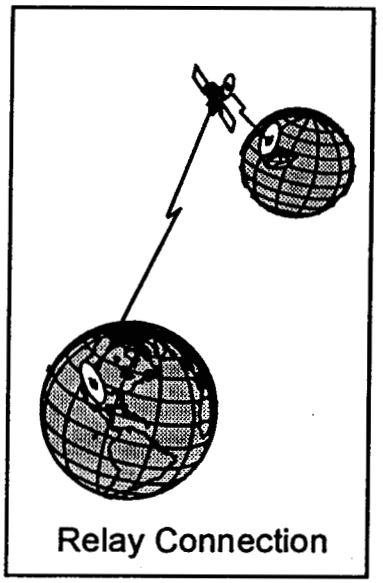
MULTI-HOP SCENARIO
(store and forward)

When a transaction is received that is addressed to a user, not serviced by receiving RTP agent, the RTP manager will determine if it should partake in the transfer. If its decision is to accept the role of participant in the transfer; the file will be assembled and placed in the local filestore and the required reports will be communicated to the sending RTP agent. The RTP manager will queue a transfer request for the transaction, and initiate a forwarding session via its dispensing entity when appropriate. The "dispensing entity" becomes a forwarding agent and transfers the transaction's data set to the next agent on its journey to the designated recipient. In each link the receiving RTP agent will accept the incoming file and report on the received data, as required, with its companion sending RTP agent. When the transfer is complete, an acknowledgment can be sent to the originating RTP by the recipient by sending a report which indicates that it is also to be forwarded to the originating RTP agent. Scenario picture 2 illustrates the store and forward operation.

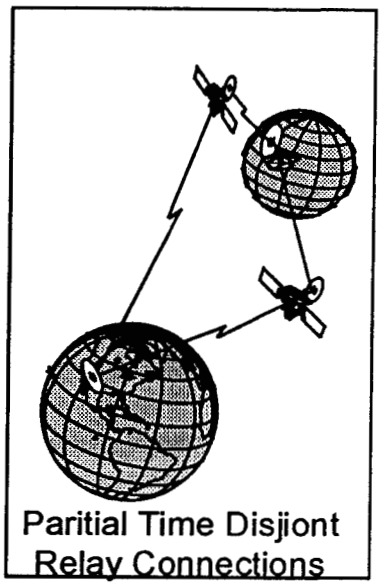
The sending RTP agent can complete a transaction using a series of "reliable RTP agents" functioning in the store and forward mode. A "reliable agent" is a RTP agent that can be counted on to get the portion of the transaction that it has acknowledged to its destination. In this case the destination RTP agent can receive portions of the same transaction from multiple RTP intermediate agents and will integrate the data as required. This functionality is herein call partial custody transfer because each receiving RTP accepts the responsibility to forward the portion of the transaction data it received to the destination. The use of multiple intermediate reliable agents is depicted in scenario picture 3. The pass for each orbiter is insufficient for a complete file transfer and the receiving entity must receive each transaction partial separately and merges them before notifying the designated application of the files arrival..



Scenario Picture 1



Scenario Picture 2



Scenario Picture 3